**Eng. 100: Music Signal Processing** W24

©J. Fessler
December 17, 2023 11:33

# Lab 3: Discrete spectral analysis

## 1 Abstract

This lab will teach you about a central tool in signal processing: using the discrete Fourier transform (DFT) as computed by the fast Fourier transform (FFT) to perform spectral analysis of signals. The Background section presents: (a) the physics of vibrating strings; (b) the Fourier series expansion of a periodic signal; (c) the sampling theorem; and (d) computation of Fourier series coefficients. The goals of this lab are: (1) To learn to use `Julia`'s `fft` function to compute Fourier series coefficients from samples of a periodic signal; (2) To use these computed coefficients to filter a noisy signal; (3) To compute these coefficients for simple sinusoids and plucked string signals to determine their spectra. These methods have important applications in the music synthesizer and transcriber projects, and elsewhere.

## 2 Background

Presenting this material thoroughly requires a lot of mathematics. There are three separate appendices that give the details at the end of this lab. This means you can skip them on first reading of the lab, and then read through them later. You will not be tested on the derivations, but if you approach college with the strategy of learning only what might be on the test, you will miss a lot.

Appendix A is a simple 1-D analysis of the physics of a vibrating string. You should be able to follow it based on the basics of the first month of Math 115 (differential calculus). The point of this material is that there are physical reasons why a vibrating string sounds like a fundamental sinusoid plus higher harmonics. Similar principles apply to other musical instruments, and thus to music in general. Daniel Bernoulli (1700-1782) first proposed this model for music, and we will follow him.

Appendix B derives the sampling theorem for band-limited periodic signals: A signal can be reconstructed perfectly from its samples if it is sampled at greater than twice its maximum frequency. This amazing result is why digital signal processing, CDs, DVDs, and pretty much everything digital exists: without it, digital could never be as good as analog!. In fact, the assumption of periodicity is unnecessary, but the proof for non-periodic signals requires much more math, specifically, the Fourier transform. Assuming periodicity is not overly restrictive because we could assume the period is one century or more, in which case we will not be around when it starts to repeat! So the question of whether a given signal is periodic or not becomes somewhat moot if we consider a long period.

Appendix C derives explicit formulas for computing the Fourier coefficients from sampled data. The math here is just algebra and trigonometric identities, so it not nearly as bad as it looks (really!), but the intermediate formulas are quite long. But it sure beats solving a huge set of simultaneous linear equations! These are the formulas we will use for spectral analysis of music.

## 3 Fourier series expansions of periodic signals ("All You Need is Trig")

From Appendix A, we can model the sound of a plucked string (here $T = 2L/a$) in either of these forms:

$$\begin{aligned} x(t) &= C_0 + \sum_{k=1}^{M} C_k \cos(2\pi kt/T - \theta_k) && \text{“sinusoidal Fourier series”} \\ &= A_0 + \sum_{k=1}^{M} A_k \cos(2\pi kt/T) + B_k \sin(2\pi kt/T) && \text{“trigonometric Fourier series,”} \end{aligned} \tag{1}$$

for some constants $\{A_k, B_k, C_k, \theta_k\}$ related to each other as follows (recall Lab 1 Appendix):

$$A_k = C_k \cos\theta_k, \quad B_k = C_k \sin\theta_k, \quad C_k = \sqrt{A_k^2 + B_k^2}, \quad \tan\theta_k = B_k/A_k. \tag{2}$$

What does $x(t)$ look like? Clearly it is periodic (it repeats in time) with period $= T$:

$$x(t) = x(t+T) = x(t+2T) = \cdots = x(t-T) = x(t-2T) = \cdots .$$

In fact *any* periodic function that occurs in the real world can be represented as a sum of sinusoids with frequencies that are integer multiples of the "fundamental frequency" $= 1/$period of the function. We need to include the integer zero here because the signal may have a DC (constant) component. But music signals do not have DC components[1], so in the sequel we assume there is no DC component, *i.e.*, we will set $A_0 = C_0 = 0$. Also note that $B_0 = 0$ because $\sin(0) = 0$.

If we let $M \to \infty$ then we can handle non-real-world signals like square waves. However, this would get into complicated mathematical convergence issues. In 1807, at a meeting of the Paris Academy, Joseph Fourier claimed any periodic function could be expanded as a (possibly infinite) sum of sinusoids. Joseph-Louis Lagrange stood up and said he was wrong, and an argument ensued that lasted for years (they did not have TV then). It turns out that there are exceptions, but they are (as mathematicians put it) "pathological," so do not worry about them. We will use finite values for $M$ here.

What does this mean for Engin 100? Musical signal segments are periodic and can be expressed as a weighted sum of sinusoidal signals whose frequencies are integer multiples of the "frequency"$=1/$period of the musical signal. If there is only one sinusoid, then the musical signal is a pure sinusoidal tone. If there is more than one sinusoid, then the set of sinusoids are called harmonics of the fundamental sinusoid, which would be the pure tone ($T =$ period):

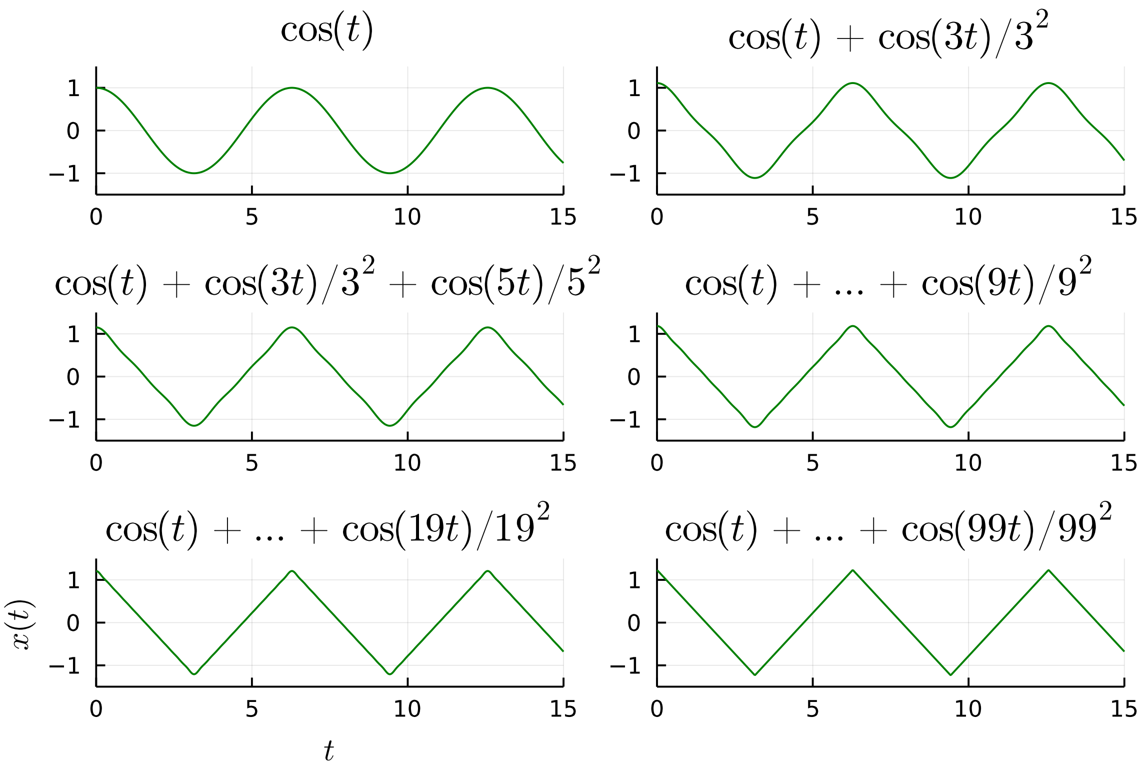
$$x(t) = \underbrace{C_1 \cos(2\pi t/T - \theta_1)}_{\text{Fundamental = 1st Harmonic: } \frac{1}{T}\text{ Hz}} + \underbrace{C_2 \cos(4\pi t/T - \theta_2)}_{\text{2nd Harmonic: } \frac{2}{T}\text{ Hz}} + \underbrace{C_3 \cos(6\pi t/T - \theta_3)}_{\text{3rd Harmonic: } \frac{3}{T}\text{ Hz}} + \cdots . \tag{3}$$

- The constants $C_k$ determine the timbre (pronounced "tam-ber") of the sound.
- The harmonics are also called partials in music circles. They change frequency if the period changes.
- We can compute $\{A_k, B_k, C_k, \theta_k\}$ easily from samples of the signal $x(t)$.

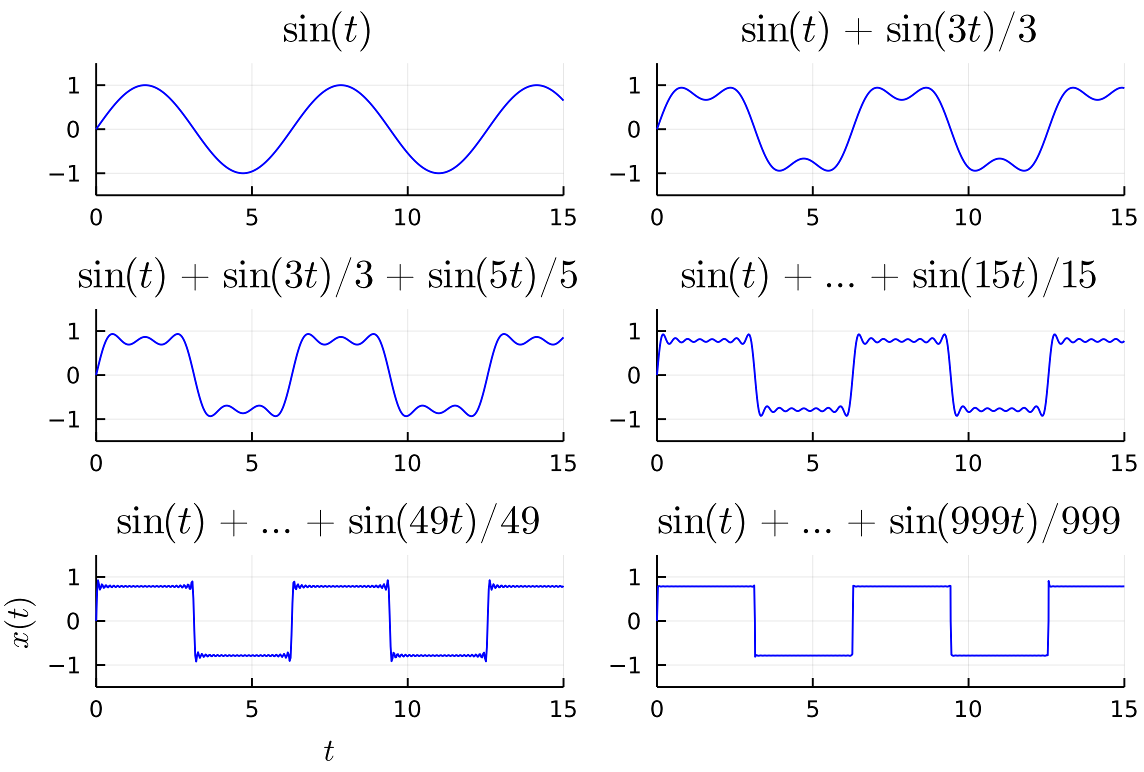
The figures on the next page show two simple examples of Fourier series converging to triangle and square waves as $M$ increases. The series for the triangle wave converges faster than that for the square wave, because triangle waves are continuous whereas square waves have discontinuities. (Such discontinuities are among the details that Fourier and Lagrange debated.)

[1]Our ears cannot hear a DC (constant) pressure level. We experience DC pressure as discomfort or pain in the ear, not as sound. Think sitting on the bottom of the deep end of a swimming pool, or cabin pressure in an airplane.

Triangle wave:



Square wave:



## 4 Using `Julia` to compute Fourier series coefficients

This lab uses the `fft` function in the `FFTW` package to compute the discrete Fourier transform (DFT) of a signal, so always begin by typing `using FFTW`.

Suppose `x` is a vector of $N$ samples of a (real, *i.e.*, not complex) signal $x[n]$. Then typing

`N = length(x); Xf = 2/N * fft(x)`

produces an output vector `Xf` that is $N$ complex numbers that are proportional to the $N$-point DFT of $x$. Assuming $N$ is even, these DFT values are related to the Fourier coefficients in (1) as follows.

| Julia | Math | Notes |
|---|---|---|
| `Xf[1]` | $2A_0$ | $(B_0 = 0)$ |
| `Xf[2]` | $A_1 - \imath B_1$ | |
| `Xf[3]` | $A_2 - \imath B_2$ | |
| ⋮ | | |
| `Xf[k]` | $A_{k-1} - \imath B_{k-1}$ | |
| `Xf[k+1]` | $A_k - \imath B_k$ | |
| ⋮ | | |
| `Xf[N/2]` | $A_{N/2-1} - \imath B_{N/2-1}$ | |
| `Xf[N/2+1]` | $2A_{N/2}$ | $(B_{N/2} = 0)$ |
| `Xf[N/2+2]` | $A_{N/2-1} + \imath B_{N/2-1}$ | (you |
| `Xf[N/2+3]` | $A_{N/2-2} + \imath B_{N/2-2}$ | can |
| ⋮ | | ignore |
| `Xf[N-1]` | $A_2 + \imath B_2$ | all of |
| `Xf[N]` | $A_1 + \imath B_1$ | these) |

In this table, $\imath = \sqrt{-1}$. In `Julia`, the built-in variable `im` equals $\sqrt{-1}$.

The `fft` function is named after the acronym FFT which is short for fast Fourier transform. The FFT is a very efficient algorithm for computing $A_k$ and $B_k$, especially when $N$ is a power of two.

Based on the table above, here is how we use the `fft` to compute the Fourier series coefficients in `Julia` given a vector `x` containing $N$ samples of a real signal over a complete period, assuming $N$ is even.

```julia
using FFTW
N = length(x)
Xf = 2/N * fft(x)
A0 = real(Xf[1]) / 2 # (This will be zero for audio signals.)
B0 = 0
A = zeros(N÷2) # To get the character "÷" type \div then hit <tab>
B = zeros(N÷2)
A[1:(N÷2-1)] = real(Xf[2:(N÷2)])
B[1:(N÷2-1)] = -imag(Xf[2:(N÷2)])
A[N÷2] = real(Xf[N÷2+1]) / 2
B[N÷2] = 0 # (redundant because we initialized B to be zero)
C0 = A0 # (This will be zero for audio signals.)
C = sqrt.(A.^2 + B.^2) # (Compute C_k for k=1,...,N/2.)
theta = atan.(B, A) # (Compute θ_k for k=1,...,N/2.)
```

- `fft(x)` has the same length as `x`.
  Similarly, there are $N/2+1$ important $A_k$ values and $N/2-1$ possibly nonzero $B_k$ values, for a total of $N$ coefficients.
- The second half of `fft(x)` is redundant with the first half (for a real signal).
  So when plotting spectra computed using `fft(x)`, *only plot the first half*!
- Then `x=N/2*real(ifft(Xf))` computes `x` from `Xf = 2/N*fft(x)`, *i.e.*, it computes the discrete-time Fourier series (3).
- The first value of `1/N * fft(x)` is the DC (average) value of `x`. For musical signals this is zero.
- `Julia` array indexes start at 1 (by default) whereas the Fourier series coefficients start with 0, *i.e.*, $A_0$ and $B_0$. So be careful with the $k-1$ and `k+1` used above[2].

## 4.1 Simple example: Interpretation of `Julia`'s `fft` output

A data acquisition system provides data sampled at $1024\frac{\text{Sample}}{\text{Second}}$. The data is read into `Julia` (*e.g.*, using `wavread`) into a vector `x`. How do we use `Julia` to determine the spectrum of this signal?

Suppose we use the following `Julia` statements, and get the following results:
- `N = length(x)` yields 3072. The duration is: (3072 samples) $\left(\frac{1}{1024}\frac{\text{Second}}{\text{Sample}}\right) = T = 3$ seconds.
- `-(2/N)*imag(fft(x))` yields a vector of 3072 numbers that are $< 10^{-14}$.
  That is numerical roundoff error, meaning these values are zero in this case!
- `(2/N)*real(fft(x))` yields a vector of mostly zeros, *except*:
  at array indexes `97,193,289,2785,2881,2977`, the values are `4,3,2,2,3,4`, respectively.
  (We can find those array indexes automatically using the `findall` function.)
- Given all of the above information, determine a sum-of-sinusoids formula for the data.

We interpret these results as follows:
- The duration $= T = 3$ seconds means we take the periodic extension of the 3-second-long data.
- This models the data as being periodic with period $T=3$ seconds, with harmonics at $\frac{1}{3}, \frac{2}{3}, \ldots$ Hertz.
- The final three components are the "mirror images" of the first three, both in value and location:
  $2785 = 3072+2-289$;
  $2881 = 3072+2-193$;
  $2977 = 3072+2-97$.
  (Why do we need to add 2 each time?)
- A component at `Julia` array index `l` corresponds to frequency $(l-1)\frac{S}{N} = (l-1)\frac{1}{T}$ where $N = ST$.
  Think "$k = l-1$" where $k$ is the mathematical index in (1) and (2) and `l` is the `Julia` array index.
  If it helps you remember this, think about the alphabetical order of $k$ and `l`, and also note that the letter `l` looks a lot like the number "`1`" and in `Julia`, array indexes start from `1`.
- Here we have components at $(97-1)/3 = 32$ Hertz, $(193-1)/3 = 64$ Hertz, and $(289-1)/3 = 96$ Hertz.
  Those are the only frequency components!
- These are all pure cosines, because the $B_k$ values are zero (to within roundoff as mentioned above).
- Thus we can write the signal as the following model:

$$x(t) = 4\cos(2\pi 32 t) + 3\cos(2\pi 64 t) + 2\cos(2\pi 96 t)\,.$$

- In this case, the signal happens to be periodic with fundamental period $T_0 = 1/32$ second, not 3 seconds.

[2] `Julia` has a package called FFTViews that is helpful for more complicated FFT-related operations. You are welcome to use it if you want, but it will likely be more work than simply adding or subtracting 1 as needed here.

### 4.2 Harder example: Interpretation of Julia's fft output

Let's examine the spectrum of a sampled train whistle signal.

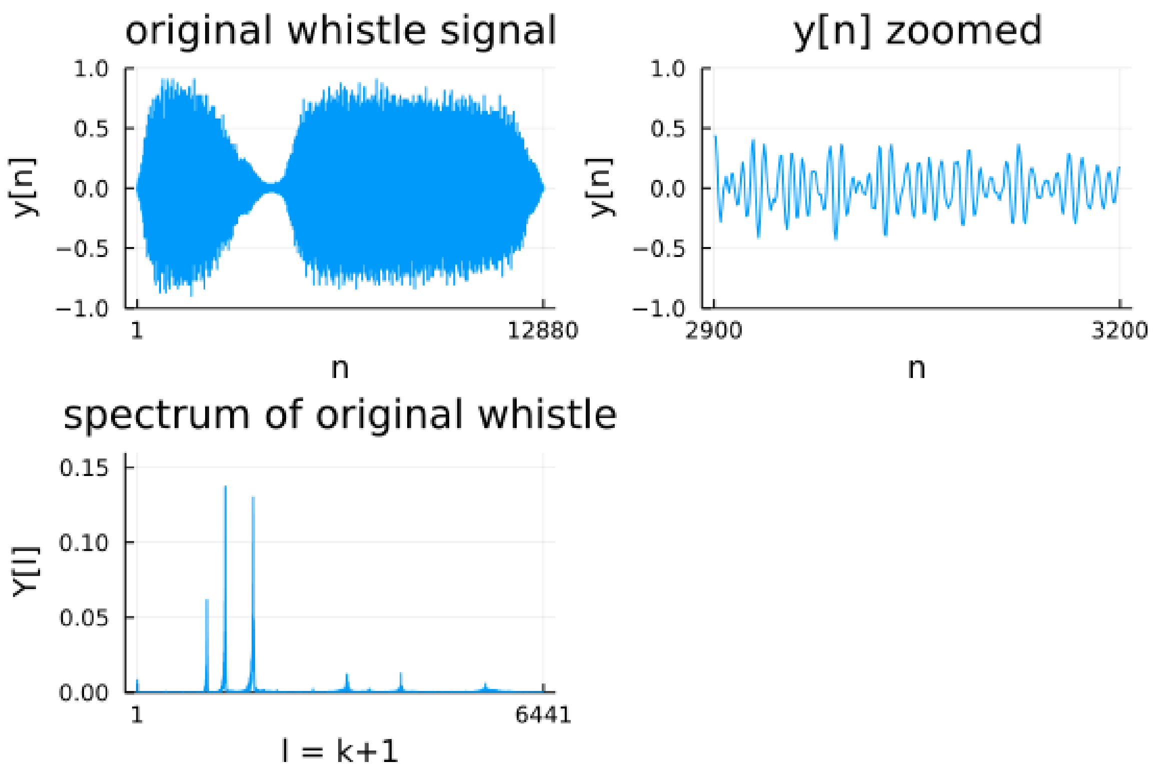
We run the following Julia statements, and get the following results.

```julia
using HTTP
using FileIO: load
using FFTW: fft
#using WAV
using Plots; default(label="")
url = "https://web.eecs.umich.edu/~fessler/course/100/misc/train-whistle.wav"
y, S, _, _ = load(HTTP.URI(url)) # read train whistle signal
N = length(y)
p1 = plot(y, xlabel="n", ylabel="y[n]", ylim=(-1,1),
    xtick=([1,N], ["1", "$N"]), title="original whistle signal")

nlim = 2900:3200
p2 = plot(nlim, y[nlim], xlabel="n", ylabel="y[n]", ylim=(-1,1),
    xtick=nlim[[1,end]], title="y[n] zoomed")

Y = 2/N * fft(y) # compute spectrum
p3 = plot(abs.(Y[1:N÷2]), xlabel="l = k+1", ylabel="Y[l]", ylim=(0,0.16),
    xtick = [1,N/2+1], title = "spectrum of original whistle")

plot(p1, p2, p3); # savefig("train1.png")
```



play

- Remember to plot only the first half of `abs.(Y)` to get the spectrum! (This is a very common mistake.)

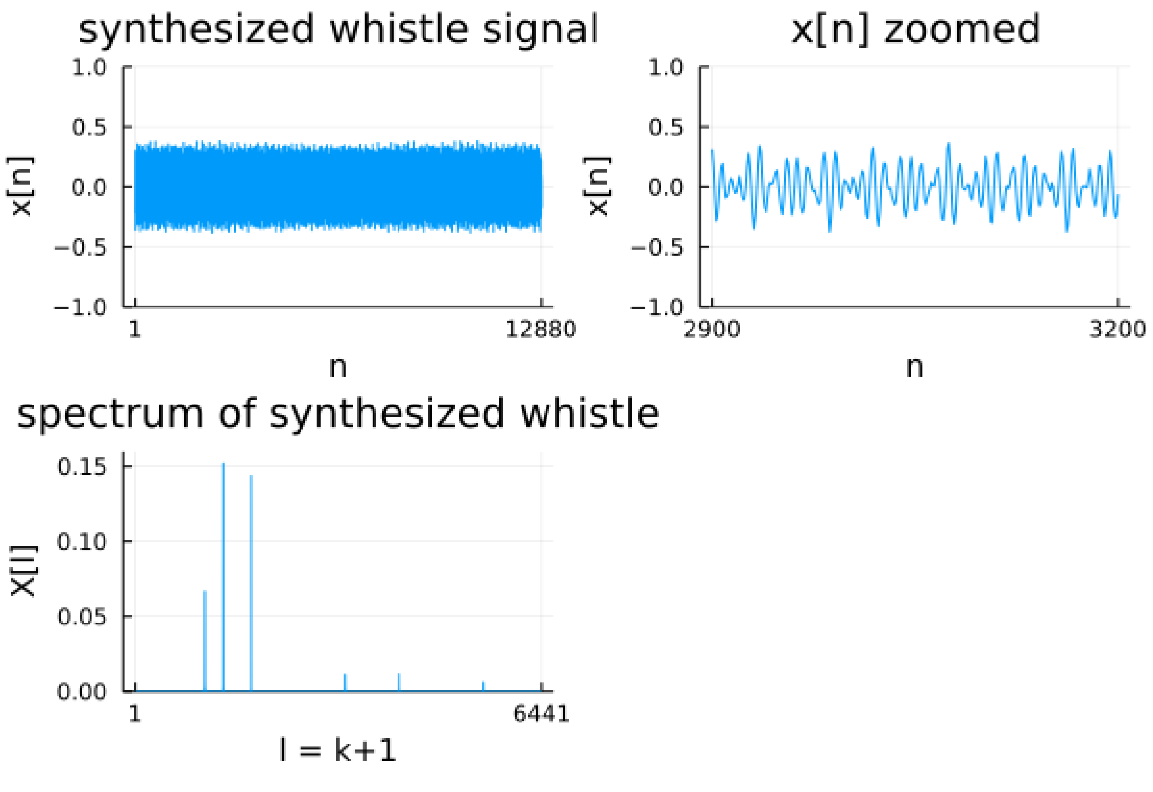
- `N = length(y)` is 12880. The duration of the signal is: $T = N/S = \frac{12880\,\text{samples}}{8192\,\frac{\text{Sample}}{\text{Second}}} = 1.57$ seconds.
- The sinusoid depicted by a spike at Julia index `l` has frequency $\frac{k}{T} = \frac{k}{N} S$ Hz, where $k = l - 1$.
- We can compute the amplitudes of each sinusoidal component using `abs.(Y[2:div(N,2)])` and the phases using `angle.(Y[2:div(N,2)])`
- Although the spectrum has many nonzero values, there are only 6 "peaks" that are significantly larger than zero:

| Array index | `l` | 1108 | 1394 | 1839 | 3326 | 4181 | 5521 |
|---|---|---|---|---|---|---|---|
| frequency [Hertz] | $\frac{l-1}{N} S$ | 704 | 886 | 1169 | 2115 | 2659 | 3511 |

You now know the frequencies, and you can read off the amplitudes from the plot. The signal is approximately

$$x(t) = 0.0684\cos(2\pi 704t - 1.70) + 0.152\cos(2\pi 886t + 2.93) + 0.144\cos(2\pi 1169t + 1.35)$$
$$+ 0.0137\cos(2\pi 2115t + 1.41) + 0.0144\cos(2\pi 2659t + 0.84) + 0.0066\cos(2\pi 3511t + 3.03)\,.$$

- The first 3 frequencies are in 5:3 and 5:4 ratios. This is suggestive of chords in the whistle sound.
- The 4th-6th frequencies are triple the 1st-3rd frequencies. Are these harmonics? It seems so.
- The following figure shows the synthesized signal and its spectrum. The original signal in fact was not periodic, so the synthesized version is only an approximation. The synthesized signal would have produced a better match if we had selected only a *segment* of the original signal.



play

This approach of approximating a signal using sinusoids is the basic idea behind MP3 audio compression and JPEG image compression: store 18 numbers instead of 12880! For more details, see [1].

## 5 Lab 3: What you have to do

1. [13] Examining spectra of simple signals

   Combine the following three subplots into one figure, and answer each question.
   (a) [1] Type the following.

   ```
   using Plots; default(label="")
   using FFTW: fft
   N=500; S=500; x = 3*cos.(2pi * 137 * (0:N-1) / S)
   X = fft(x); plot(2/N*abs.(X), xlabel="frequency index l=k+1", ylabel="|X[l]|")
   ```

   Examine values of `abs.(X)` to confirm that this is a sinusoid with frequency 137 Hertz.
   It might be helpful to use `findall(2/N*abs.(X) .> 1e-8)`
   - [1] What is the amplitude of this sinusoid?
   - [1] How does its amplitude appear in the plot?

   Note the second peak is a mirror image of the first! Indeed, we just made the "common mistake" mentioned earlier.
   In this case of course we know that the original signal `x` is a 137 Hz sinusoid, so the spectrum is what we expected. But if someone gave you the signal sample vector `x` without telling you the frequency, you can still use a plot of `abs.(2/N*fft(x))` to determine the frequency and amplitude.

   (b) [1] Type the following. (Notice the small change!)

   ```
   using Plots; default(label="")
   using FFTW: fft
   N=500; S=500; y = 3*cos.(2pi * 137.1 * (0:N-1)/S)
   Y = fft(y); plot(2/N*abs.(Y), xlabel="frequency index l=k+1", ylabel="|Y[l]|")
   ```

   Note the broader base around the peak.
   Examine the plot to determine the *approximate* frequency of this sinusoid.
   - [2] Why don't you get just a sharp spike or two, like you did previously?
     Hint: Is the signal sampled over an integral number of periods?

   (c) Download `plucked_synth.mat` from Canvas.
   This is a *synthetic* signal sampled at $8192 \frac{\text{Sample}}{\text{Second}}$.
   - [1] Type the following

   ```
   using MAT: matread
   using Sound: soundsc
   using FFTW: fft
   using Plots; default(label="")
   #plotly() # if this fails, then comment it out
   file = "../../../data/plucked_synth.mat" # edit path as needed
   z = vec(matread(file)["y"]) # read
   N = length(z)
   soundsc(z, 8192)
   Z = fft(z); plot(2/N*abs.(Z), xlabel="frequency index l=k+1", ylabel="|Z[l]|")
   xlims!(1,1+N÷2) # just show left half of spectrum
   ```

   - [4] What is its period and fundamental frequency?
   - [2] How many harmonics does it have?

2. [13] Reducing noise in noisy data using filtering.

   Combine the following three subplots into one figure.

   (a) [1] Download `noisy1.mat`. Type the following.

   ```
   using MAT: matread
   using Sound: sound
   using Plots; default(markerstrokecolor=:auto, label="")
   file = "../../../data/noisy1.mat" # edit path as needed
   y = vec(matread(file)["y"]) # read
   sound(y, 8192)
   plot(y, xlabel="n", ylabel="y[n]", title="Lab3 5.2a: noisy signal")
   ```

   (It should look like random noise.)

   (b) [2] Plot the spectrum of this noisy signal.

   (c) [10] Hidden in this noise is a signal, sampled at $1000 \frac{\text{Sample}}{\text{Second}}$.
   All you know is that: it is periodic with period=0.2 seconds, and it is band-limited to about 20 Hz.
   From that information, clean up the signal by filtering out most of the noise.
   - Hints: How many harmonics does it have, and at what frequencies?
   - Set other frequencies to zero and use `x = real(ifft(X))` where `X=fft(x)`.
     Hint: if `z = [20, 30, 40, 50, 60]` and we type `z[[2; 4:5]] .= 0` and then look at `z` again, we get `z = [20, 0 40, 0, 0]`.

   Plot the cleaned-up signal $x[n]$.

3. [8] Spectrogram: Computing separate spectra of different intervals of a single signal

   Combine the following three subplots into one figure, and answer each question.

   (a) Chirp signal

   - [1] Type `x = cos.((1:1000).^2/1000); plot(x)`
     [1] Describe this signal. (Try listening to it.)
   - [1] Type the following.

   ```
   using FFTW: fft
   using Plots
   N = 1000; S = 1000; x = cos.((1:N).^2/S)
   N2 = 40
   x2 = reshape(x,N2,:) # 25 segments of N samples
   X2 = fft(x2, 1) # 1D fft of each column of x2
   heatmap(2/N2 * abs.(X2), xlabel="time segment", ylabel="l=k+1") # spectrogram
   ```

     [0] See what this is doing?

   (b) Tonal music

   - Download the file `victors_tone.wav` from Canvas.
   - [1] Type the following.

   ```
   using WAV: wavread
   using FFTW: fft
   using Plots: heatmap
   file = "../../../data/victors_tone.wav" # adjust as needed
   (y, S) = wavread(file)
   N2 = 300
   y2 = reshape(y, N2, :) # 260 segments of length N
   Y2 = fft(y2, 1) # 1D fft of each column of y2
   heatmap(2/N2 * abs.(Y2), xlabel="time segment", ylabel="l=k+1") # spectrogram
   ```

     [0] See what this is doing?
   - [2] Can you read off the relative pitches and durations of the tones from this spectrogram?
   - [2] Could a spectrogram of a musical signal serve as a type of musical notation?

4. [16] Removing interference from a signal

   A Michigan State fan broke into the Engin 100 Canvas web site and corrupted “The Victors” by adding the MSU fight song to it! How can you remove this interference to undo this heresy?

   (a) Spectrum

   - Download the file `victors_msu.wav` from Canvas. Type the following to show part of its spectrum.

   ```
   using WAV: wavread
   using Sound: sound
   using FFTW: fft
   using Plots; default(label="")
   file = "../../../data/victors_msu.wav" # adjust as needed
   (z, S) = wavread(file)
   #sound(z, S) # uncomment this to hear a heresy
   N = length(z)
   Z = fft(z)
   plot(2/N * abs.(Z), xlims=(1,17000), xlabel="frequency index l=k+1", ylabel="Z[l]")
   ```

- [1] Can you distinguish the UM and MSU fight song spectra?

(b) Spectrogram

- [1] Type the following.

```
N2 = 300
z2 = reshape(z, N2, :) # 260 segments of length N2
Z2 = fft(z2, 1) # 1D fft of each column of z2
heatmap(2/N2 * abs.(Z2), xlabel="time segment", ylabel="l=k+1") # spectrogram
```

- [1] How about now? It might help to try `ylims!(1,50)` to zoom in some.

(c) Victory

- [12] Type `G = copy(Z)` and then set some values of `G` to zero so that `G` only has "good" stuff in it. Then type `g = real(ifft(G)); sound(g, S)`
- Display the spectrogram of `g`
- Show your lab instructor that you successfully "eliminated" MSU, both with audio and graphically. Explain your calculations in the lab report.

## 6 Report

Just like in Lab 2, put all your plots and answers to the questions into a single document and upload to Gradescope as a pdf file. The audience is again your lab instructor.
To keep on schedule, you should finish this before the start of the lab where your lab section begins Project 2.

RQ Lab3.1. A periodic signal has period $T = 0.002$ seconds. What is the frequency of its 2nd harmonic (in Hz)?

RQ Lab3.2. A sinusoidal signal `x` has 16384 samples acquired at 8192$\frac{\text{Sample}}{\text{Second}}$. The statement `plot(fft(abs.(x)))` shows two peaks, the first of which is at `Julia` index 441. What is the frequency of the sinusoid?

## 7 Appendix A: Physics of vibrating strings

The basic instrument of rock and folk music is the guitar. A guitar has several strings (usually 6); plucking these strings makes them vibrate and creates sound. Let us examine the basic physics of a horizontal vibrating string. This requires basic differential calculus that you should have seen by now in Math 115 or equivalent.

Let $x$ denote horizontal position along a string of length $L$ and let $y(x,t)$ denote the vertical displacement of the string at position $x$ and time $t \geq 0$. Let $T$ denote the tension in the string (which the player adjusts by the tuning knobs). The tension $T$ does not vary with $x$ because the change of length of the string is negligible when it is plucked or vibrating.

Recall that the slope of a function $y(x)$ at position $x$ is $\frac{dy}{dx}$. The vertical component of force at position $x$ is $T\sin(\theta(x)) \approx T\tan(\theta(x)) = T\frac{dy}{dx}$ and the vertical component of force at position $x+\Delta$ is $T\frac{dy}{dx} + \frac{d}{dx}[T\frac{dy}{dx}]\Delta$ in the opposite direction. The net force is the difference $T\frac{d^2y}{dx^2}\Delta$, because $T$ is assumed not to vary with $x$.

This force acts on a mass $\rho\Delta$, where $\rho$ is the linear density (mass per unit length) of the string. Newton's law of motion $F = ma$ in the vertical direction then gives

$$ma = (\rho\Delta)\frac{d^2y}{dt^2} = F = T\frac{d^2y}{dx^2}\Delta \Longrightarrow \frac{d^2y}{dt^2} = a^2\frac{d^2y}{dx^2},$$

| Quantity: | $T$ | $\rho$ | $a^2 = T/\rho$ |
|---|---|---|---|
| Units: | mass $\frac{\text{length}}{\text{time}^2}$ | $\frac{\text{mass}}{\text{length}}$ | $\frac{\text{length}^2}{\text{time}^2}$ |

where $a^2 = T/\rho$ depends only on the string material itself. The derivatives here should really be partial derivatives $\frac{\partial^2 y}{\partial t^2}$ and $\frac{\partial^2 y}{\partial x^2}$; this just means that the other variable is treated as a constant in each case.

Because the ends of the string are fixed at both ends $x = 0$ and $x = L$, the differential equation and its solution are

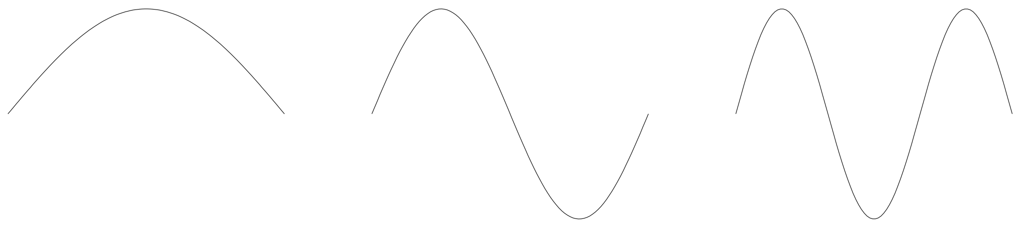
$$\frac{d^2y}{dt^2} = a^2\frac{d^2y}{dx^2}; \quad y(x=0,t) = y(x=L,t) = 0 \Longrightarrow y(x,t) = \sum_{k=1}^{M} E_k \sin(k\pi x/L)\cos(ak\pi t/L - \theta_k)\,. \tag{4}$$

You can differentiate the solution (4) to verify that it solves the differential equation. (You need a course in partial differential equations to find such solutions in general.)

What does (4) *look* like? If we took a snapshot of the vibrating string at fixed time $t = t_0$ we would see

$$y(x,t_0) = \sum_{k=1}^{M} E_k \cos(ak\pi t_0/L - \theta_k)\sin(k\pi x/L)\,.$$

This is a weighted sum of sinusoidal functions of $x$, the first three of which look like the following (note that the endpoints are always fixed):



These components of the solution are called modes of vibration. Guitar players can emphasize different modes by placing their fingers in certain spots on the strings.

What does (4) *sound* like? Sound is generated by the time variation, which is also sinusoidal in $t$:

$$y(x_0,t) = \sum_{k=1}^{M} E_k \sin(k\pi x_0/L)\cos(k\pi at/L - \theta_k)\,.$$

## 8 Appendix B: Sampling theorem

Suppose $x(t)$ is real-valued with period $= T$ seconds and maximum frequency $\frac{M}{T}$ Hertz for some integer $M$. Then $x(t)$ can be expanded as (1). This means that $x(t)$ is specified completely by $2M+1$ constants $\{A_k, B_k\}$ or equivalently $\{C_k, \theta_k\}$. If we can determine those $2M+1$ constants, then we know $x(t)$ for all $t$. Now suppose we sample $x(t)$ at rate $\frac{N}{T}\frac{\text{Sample}}{\text{Second}}$, *i.e.*, with sampling interval $\Delta = T/N$, for some $N \geq 2M+1$, obtaining the samples

$$x[n] = x(t)\Big|_{t=n\Delta} = x\left(n\frac{T}{N}\right), \quad n = 1, \ldots, N.$$

Setting $t = \frac{nT}{N}$, $n = 1, \ldots, N$ in (1) yields $N$ linear equations in $2M+1$ unknowns $\{A_0, A_k, B_k,\ k = 1, \ldots, M\}$:

$$x[n] = x(nT/N) = A_0 + \sum_{k=1}^{M} A_k \cos((2\pi k/T)(nT/N)) + B_k \sin((2\pi k/T)(nT/N)), \quad n = 1, \ldots, N.$$

If $N \geq 2M+1$ then we can solve these equations to obtain the coefficients $\{A_0, A_k, B_k,\ k = 1 \ldots M\}$ from samples $\{x(nT/N),\ k = 1 \ldots N\}$. So we can reconstruct $x(t)$ *exactly* from its samples $x(nT/N)$. Note that we need to sample $x(t)$ at a rate $\frac{N}{T}\frac{\text{Sample}}{\text{Second}}$, slightly more than *double* the maximum frequency $\frac{M}{T}$ Hertz. The doubling is because we need *pairs* of constants $\{A_0, A_k, B_k,\ k = 1, \ldots, M\}$. This sampling theory was developed by UM alumnus Claude Shannon. (That's his bust outside the EECS building.)

### 8.1 Small illustrative numerical example

Let $x(t)$ be periodic with period $T = 0.001$ s and have maximum frequency 1000 Hertz, *i.e.*, $M = 1$.
Suppose $x(t)$ is sampled at $4000\frac{\text{Sample}}{\text{Second}} = 4/T$, so $N = 4 \geq 2M+1$. The goal is to reconstruct a formula for the signal.
We observe: $x(1/4000) = 40$; $x(2/4000) = 10$; $x(3/4000) = 20$; $x(4/4000) = 50$. Because the signal is periodic:

$$x(t) = a_0 + a_1 \cos(2\pi t/0.001) + a_2 \cos(4\pi t/0.001) + \cdots + b_1 \sin(2\pi t/0.001) + b_2 \sin(4\pi t/0.001) + \ldots.$$

Because the signal has maximum frequency 1000 Hertz, $x(t) = a_0 + a_1 \cos(2\pi t/0.001) + b_1 \sin(2\pi t/0.001)$.
So the signal is completely determined by three constants: $\{a_0, a_1, b_1\}$. Substituting in $t = n/4000$ for $n = 1, 2, 3, 4$ gives

$$\begin{aligned}
40 = x\left(\frac{1}{4000}\right) &= a_0 + a_1 \cos\left(2\pi\frac{1}{0.001}\frac{1}{4000}\right) + b_1 \sin\left(2\pi\frac{1}{0.001}\frac{1}{4000}\right) = a_0 + a_1\cos\left(\frac{1\pi}{2}\right) + b_1 \sin\left(\frac{1\pi}{2}\right) = a_0 + b_1\\
10 = x\left(\frac{2}{4000}\right) &= a_0 + a_1 \cos\left(2\pi\frac{1}{0.001}\frac{2}{4000}\right) + b_1 \sin\left(2\pi\frac{1}{0.001}\frac{2}{4000}\right) = a_0 + a_1\cos\left(\frac{2\pi}{2}\right) + b_1 \sin\left(\frac{2\pi}{2}\right) = a_0 - a_1\\
20 = x\left(\frac{3}{4000}\right) &= a_0 + a_1 \cos\left(2\pi\frac{1}{0.001}\frac{3}{4000}\right) + b_1 \sin\left(2\pi\frac{1}{0.001}\frac{3}{4000}\right) = a_0 + a_1\cos\left(\frac{3\pi}{2}\right) + b_1 \sin\left(\frac{3\pi}{2}\right) = a_0 - b_1\\
50 = x\left(\frac{4}{4000}\right) &= a_0 + a_1 \cos\left(2\pi\frac{1}{0.001}\frac{4}{4000}\right) + b_1 \sin\left(2\pi\frac{1}{0.001}\frac{4}{4000}\right) = a_0 + a_1\cos\left(\frac{4\pi}{2}\right) + b_1 \sin\left(\frac{4\pi}{2}\right) = a_0 + a_1.
\end{aligned}$$

Solving these four linear equations in three unknowns gives $a_0 = 30,\ a_1 = 20,\ b_1 = 10$. So we have found:

$$\begin{aligned}
x(t) &= 30 + 20\cos(2\pi 1000t) + 10\sin(2\pi 1000t)\\
&= 30 + \sqrt{5}\cos\left(2\pi 1000t - \tan^{-1}(1/2)\right) = 30 + 2.236\cos(2\pi 1000t - 26.6^\circ).
\end{aligned}$$

This was tedious even for a signal with only a few coefficients. Thankfully there is a general solution, described next.

## 9 Appendix C: Direct computation of Fourier series coefficients

Solving $N$ linear equations in $N$ unknowns is very time-consuming if $N$ is large. Fortunately, we can avoid this effort by using formulas we now derive. We assume $N$ is even "for simplicity" but the principles apply to odd $N$ too. Warning: grab a mug of non-decaf coffee or tea before reading!

First, recall the sine and cosine addition formulas:

$$\begin{aligned}\sin(x \pm y) &= \sin(x)\cos(y) \pm \cos(x)\sin(y)\\ \cos(x \pm y) &= \cos(x)\cos(y) \mp \sin(x)\sin(y)\,.\end{aligned}$$

Adding and subtracting these gives

$$\begin{aligned}2\cos(x)\cos(y) &= \cos(x-y) + \cos(x+y)\\ 2\sin(x)\sin(y) &= \cos(x-y) - \cos(x+y)\\ 2\sin(x)\cos(y) &= \sin(x-y) + \sin(x+y)\,.\end{aligned}$$

Second, because $\sin\left(\frac{2\pi k}{N}n\right)$ is periodic in $n$ with period $N$ and $\sin(-x) = -\sin(x)$, and $\sin(k\pi) = 0$, we have:

$$\begin{aligned}\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right) &= \sum_{n=1}^{N/2-1}\sin\left(\frac{2\pi k}{N}n\right) + \sin\left(\frac{2\pi k}{N}\frac{N}{2}\right) + \sum_{n=N/2+1}^{N-1}\sin\left(\frac{2\pi k}{N}n\right) + \sin\left(\frac{2\pi k}{N}N\right)\\ &= \sum_{n=1}^{N/2-1}\sin\left(\frac{2\pi k}{N}n\right) + \sum_{n=-N/2-1}^{-1}\sin\left(\frac{2\pi k}{N}n\right) = \sum_{n=1}^{N/2}\left[\sin\left(\frac{2\pi k}{N}n\right) - \sin\left(\frac{2\pi k}{N}n\right)\right] = 0.\end{aligned}$$

In words: an even number of equally spaced samples of a sinusoid (over a multiple of $2\pi$) sum to zero.

Now consider the case of cos:

$$\begin{aligned}&2\sin\left(\frac{2\pi k}{N}\right)\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right) = \sum_{n=1}^{N}2\sin\left(\frac{2\pi k}{N}\right)\cos\left(\frac{2\pi k}{N}n\right)\\ &= \sum_{n=1}^{N}\left[\sin\left(\frac{2\pi k}{N}(n+1)\right) - \sin\left(\frac{2\pi k}{N}(n-1)\right)\right] = \sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}(n+1)\right) - \sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}(n-1)\right) = 0.\end{aligned}$$

Thus, as long as $\sin\left(\frac{2\pi k}{N}\right) \neq 0$, *i.e.*, as long as $k \notin \{0, \pm N/2, \pm N, \pm 3N/2, \ldots\}$, we can divide by $\sin\left(\frac{2\pi k}{N}\right)$ and conclude that $\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right) = 0$. If $k = N/2$ (for even $N$) then

$$\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right) = \sum_{n=1}^{N}\cos(\pi n) = \sum_{n=1}^{N}(-1)^n = 0.$$

Similarly, if $k = N$ then

$$\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right) = \sum_{n=1}^{N}\cos(\pi 2n) = N.$$

Summarizing, for $N$ even we have

$$\begin{aligned}&\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right) = 0\\ &\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right) = \begin{cases} N, & k = 0,\ k = \pm N,\ k = \pm 2N, \ldots\\ 0, & \text{otherwise.}\end{cases}\end{aligned}$$

Third, considering products of sin and cos, we now have

$$2\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right)\cos\left(\frac{2\pi l}{N}n\right)=\sum_{n=1}^{N}\left[\sin\left(\frac{2\pi(k-l)}{N}n\right)+\sin\left(\frac{2\pi(k+l)}{N}n\right)\right]=0-0=0 \text{ even if } k=l.$$

We also now have

$$2\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right)\sin\left(\frac{2\pi l}{N}n\right)=\sum_{n=1}^{N}\left[\cos\left(\frac{2\pi(k-l)}{N}n\right)-\cos\left(\frac{2\pi(k+l)}{N}n\right)\right]=\begin{cases}0, & k\neq l\\ N, & k=l\end{cases}$$
$$2\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right)\cos\left(\frac{2\pi l}{N}n\right)=\sum_{n=1}^{N}\left[\cos\left(\frac{2\pi(k-l)}{N}n\right)+\cos\left(\frac{2\pi(k+l)}{N}n\right)\right]=\begin{cases}0, & k\neq l\\ N, & k=l.\end{cases}$$

These three equations are called orthogonality conditions, because they state that the inner products (aka dot products) of $\{\sin(\frac{2\pi k}{N}n)\}$ for two different frequency indexes $k$ are zero, and similarly for $\{\cos(\frac{2\pi k}{N}n)\}$.

Finally (hooray!), setting $t=nT/N$ in (1) gives

$$x[n]=x(t)\Big|_{t=\frac{nT}{N}}=\sum_{k=0}^{M}A_k\cos\left(\frac{2\pi k}{N}n\right)+\sum_{k=1}^{M}B_k\sin\left(\frac{2\pi k}{N}n\right). \qquad (5)$$

This is the Discrete-Time Fourier Series (DTFS). Multiplying (5) by $\sin(\frac{2\pi m}{N}n)$ and summing over $n$ gives

$$\begin{aligned}\sum_{n=1}^{N}x[n]\sin\left(\frac{2\pi m}{N}n\right)&=\sum_{n=1}^{N}\left[\sum_{k=0}^{M}A_k\cos\left(\frac{2\pi k}{N}n\right)\right]\sin\left(\frac{2\pi m}{N}n\right)+\sum_{n=1}^{N}\left[\sum_{k=1}^{M}B_k\sin\left(\frac{2\pi k}{N}n\right)\right]\sin\left(\frac{2\pi m}{N}n\right)\\&=\sum_{k=0}^{M}A_k\left[\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right)\sin\left(\frac{2\pi m}{N}n\right)\right]+\sum_{k=1}^{M}B_k\left[\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right)\sin\left(\frac{2\pi m}{N}n\right)\right]\\&=NB_m/2\end{aligned}$$

using the orthogonality conditions. Similarly, multiplying (5) by $\cos(\frac{2\pi m}{N}n)$ and summing over $n$ gives

$$\begin{aligned}\sum_{n=1}^{N}x[n]\cos\left(\frac{2\pi m}{N}n\right)&=\sum_{n=1}^{N}\left[\sum_{k=0}^{M}A_k\cos\left(\frac{2\pi k}{N}n\right)\right]\cos\left(\frac{2\pi m}{N}n\right)+\sum_{n=1}^{N}\left[\sum_{k=1}^{M}B_k\sin\left(\frac{2\pi k}{N}n\right)\right]\cos\left(\frac{2\pi m}{N}n\right)\\&=\sum_{k=0}^{M}A_k\left[\sum_{n=1}^{N}\cos\left(\frac{2\pi k}{N}n\right)\cos\left(\frac{2\pi m}{N}n\right)\right]+\sum_{k=1}^{M}B_k\left[\sum_{n=1}^{N}\sin\left(\frac{2\pi k}{N}n\right)\cos\left(\frac{2\pi m}{N}n\right)\right]\\&=NA_m/2\end{aligned}$$

using the orthogonality conditions again.

The *bottom line* is that we have derived the following formulas for the Fourier series coefficients in terms of the signal samples:

$$\boxed{A_m=\frac{2}{N}\sum_{n=1}^{N}x[n]\cos\left(\frac{2\pi m}{N}n\right),}\qquad\boxed{B_m=\frac{2}{N}\sum_{n=1}^{N}x[n]\sin\left(\frac{2\pi m}{N}n\right),}\qquad m=1,2,\ldots,N/2. \qquad (6)$$

We can plug these coefficients into (1) to get an explicit formula for the signal $x(t)$ for all $t$.
In other words, we need not solve $N$ equations in $N$ unknowns; we can write down the solution directly, thanks to those orthogonality conditions! These formulas compute Fourier series coefficients *directly* from *sampled* data.

### 9.1 Small illustrative numerical example, continued

Continuing the example in Appendix B, we can compute the coefficients $\{a_0, a_1, b_1\}$ directly using (6):

$$\begin{aligned}
a_0 &= \frac{1}{4}\left[40\cos\left(2\pi\frac{0}{0.001}\frac{1}{4000}\right) + 10\cos\left(2\pi\frac{0}{0.001}\frac{2}{4000}\right) + 20\cos\left(2\pi\frac{0}{0.001}\frac{3}{4000}\right) + 50\cos\left(2\pi\frac{0}{0.001}\frac{4}{4000}\right)\right] = 30\\
a_1 &= \frac{1}{4}\left[40\cos\left(2\pi\frac{1}{0.001}\frac{1}{4000}\right) + 10\cos\left(2\pi\frac{1}{0.001}\frac{2}{4000}\right) + 20\cos\left(2\pi\frac{1}{0.001}\frac{3}{4000}\right) + 50\cos\left(2\pi\frac{1}{0.001}\frac{4}{4000}\right)\right] = 20\\
b_1 &= \frac{1}{4}\left[40\sin\left(2\pi\frac{1}{0.001}\frac{1}{4000}\right) + 10\sin\left(2\pi\frac{1}{0.001}\frac{2}{4000}\right) + 20\sin\left(2\pi\frac{1}{0.001}\frac{3}{4000}\right) + 50\sin\left(2\pi\frac{1}{0.001}\frac{4}{4000}\right)\right] = 10
\end{aligned}$$

in agreement with Appendix B. This process replaces solving the linear system of 4 equations in 3 unknowns. And even more importantly, a computer can do it rapidly by a fast Fourier transform (FFT) invented in 1965 [2]. It was named one of the "top 10 algorithms of the 20th century" [3].

## References

[1] C. Langton and V. Levin. *Intuitive guide to Fourier analysis and spectral estimation.* Mountcastle, 2016.

[2] J. W. Cooley and J. W. Tukey. An algorithm for machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301, April 1965.

[3] F. Sullivan and J. Dongarra. The joy of algorithms. *Computing in Science and Engineering*, 2(1):2, January 2000.